



NRL/MR/7320--06-8990

Software Development for Producing Standard Navy Surf Output from Delft3D

Y. LARRY HSU
JAMES D. DYKES
RICHARD A. ALLARD

*Ocean Dynamics and Prediction Branch
Oceanography Division*

December 29, 2006

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) 29-12-2006		2. REPORT TYPE Memorandum Report		3. DATES COVERED (From - To)	
4. TITLE AND SUBTITLE Software Development for Producing Standard Navy Surf Output from Delft3D				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER PE0603207N	
6. AUTHOR(S) Y. Larry Hsu, James D. Dykes, and Richard A. Allard				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER 73-5097-C7-5	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory Oceanography Division Stennis Space Center, MS 39529-5004				8. PERFORMING ORGANIZATION REPORT NUMBER NRL/MR/7320--06-8990	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Space and Naval Warfare Systems Command 2451 Crystal Drive Arlington, VA 22245-5200				10. SPONSOR / MONITOR'S ACRONYM(S) SPAWAR	
				11. SPONSOR / MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT The Delft3D modeling system, developed by Delft Hydraulics, is a comprehensive coastal hydrodynamic modeling system, capable of simulating hydrodynamic processes due to waves, tides, rivers, winds and coastal currents. Delft3D produces two-dimensional time-dependent forecasting output for many nearshore wave and flow parameters. But it does not produce the operational surf forecasting parameters as specified in the Joint Surf Manual. The standard surf parameters include maximum and significant breaker heights, breaker type statistics, percent of breaking, surf zone width, number of surf lines and modified surf index (MSI). Subroutines from Navy Standard Surf Model (SURF 3.2) are adapted and refined to compute these surf parameters from Delft3D output. This report describes input and output files and the software structure.					
15. SUBJECT TERMS Delft3D Wave height Surf Longshore current					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UL	18. NUMBER OF PAGES 23	19a. NAME OF RESPONSIBLE PERSON Y. Larry Hsu
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			19b. TELEPHONE NUMBER (include area code) (228) 688-5260

CONTENTS

1. INTRODUCTION.....	1
2. Delft3D OUTPUT FILES	1
3. SOFTWARE DESCRIPTION.....	2
4. SAMPLE RESULTS	5
5. FUTURE PLANS.....	6
ACKNOWLEDGEMENTS.....	6
REFERENCES	6
APPENDIX A — Sample SURF 3.2 output showing the standard Navy surf output.....	8
APPENDIX B — MATLAB delft3d2ascii.m to read Delft3D NEFIS format output files	9
APPENDIX C — Main Fortran Program surfdelft3d.f	14

1. INTRODUCTION

The one-dimensional Navy Standard Surf Model (NSSM, or SURF 3.2) has been shown to be very robust (Hsu, et al., 2002), but it can produce inaccurate wave and longshore current estimations for areas with complicated bathymetry. Since SURF assumes parallel bottom contours in the surf zone, it cannot account for longshore variations of bathymetry or forcing. Only 2D or quasi-3D nearshore models should be used for such cases. The Delft3D modeling system, developed by Delft Hydraulics (Roelvink and Banning, 1994), is a complete coastal hydrodynamic modeling system, capable of simulating hydrodynamic processes due to waves, tides, rivers, winds and coastal currents. It uses finite difference numerical techniques and can be run in Cartesian or spherical coordinates and for regular or curvilinear grids. Delft3D has been evaluated and validated by Elias et al. (2000) and Morris (2001). Recently, Hsu et al. (2006) validated the model using Duck94 and Santa Barbara data. Additional validation using SandyDuck (Duck,NC) and SURF04 (Sardinia/Italy) data sets are being conducted.

Delft3D produces two-dimensional time dependent forecasting output for many nearshore wave and flow parameters. But it does not produce the operational surf forecasting parameters as specified in the Joint Surf Manual. Using Delft3D output, this report describes the software which generates the standard surf parameters including maximum and significant breaker height, breaker type statistics, percent of breaking, surf zone width, number of surf lines and modified surf index (MSI). Subroutines from SURF 3.2 are adapted to compute these parameters.

2. Delft3D OUTPUT FILES

As described in the Delft3D-FLOW manual (Delft Hydraulics, 2005), the results of a Delft3D-FLOW computation are stored in four types of files:

- communication file: <com-runid.def> and <com-runid.dat>.
- history file: <trih-runid.def> and <trih-runid.dat>.
- map file: <trim-runid.def> and <trim-runid.dat>.
- drogue file: <trid-runid.def> and <trid-runid.dat>.

These files use the binary NEFIS format, therefore is not easily readable. MATLAB routines provided by Delft3D are used in this software development to read the data. All necessary input (cross-shore data of location, depth, significant wave height, wave angle and longshore current) to derive the standard surf parameters can be obtained from the “com” (communication) file. If the roller option is tuned on, then significant wave height needs to be derived from the wave energy term in the “trim” (map) file. When roller is on, radiation stresses and gradients of these stresses are computed in Delft3D based on the wave energy and roller energy replacing the conventional wave forces as derived from the WAVE (SWAN) model. Only the wave direction from an initial SWAN run is still used. It should be noted that the currents derived from trim file represent the Eulerian-frame (fixed location) velocity whereas those from com file represent Lagrangian-frame (following tagged water particles) velocity. But as far as longshore current is concerned, currents derived from both files are the same.

3. SOFTWARE DESCRIPTION

To help describe the objective of the software, a sample surf summary for a test case from SURF 3.2 is presented in Appendix A. The first part of the summary is the input information, and the part starting with “Coded Surf Forecast Follows” is what this software will produce from Delft3D results. Fig. 1 shows the flow chart of the MATLAB code to convert Delft3D binary output to ASCII format.

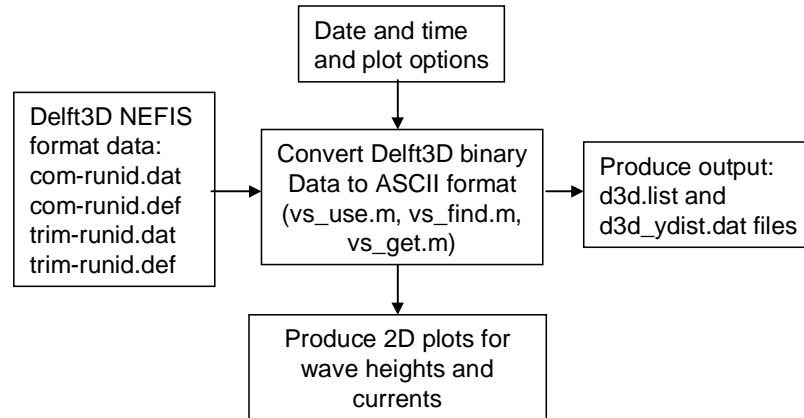


Fig. 1 – Flow chart for Matlab delft3d2ascii.m

Two types of output files are generated. The first one is called d3d.list which shows the list of data filenames at different alongshore locations. A partial sample is listed here:

d3d_0.dat
d3d_30.dat
d3d_60.dat
d3d_90.dat
.....
.....
d3d_1710.dat
d3d_1740.dat

The number in the filenames represents the longshore positions of the cross-shore data, e.g., d3d_90.dat is for data at 90-m location relative to the model grid origin. The second file consists of cross-shore data at a particular alongshore location and is partially listed below:

x	depth	wave height	longshore current	wave angle
900.00	8.26	2.34	-0.07	-7.8
890.00	8.19	2.34	-0.07	-7.8
.....				
310.00	3.83	1.92	-0.50	-3.1
300.00	3.60	1.88	-0.51	-2.7
290.00	3.32	1.83	-0.51	-2.3
280.00	2.95	1.76	-0.49	-1.7
270.00	2.50	1.64	-0.46	-1.0
.....				

Fig. 2 shows the flow chart of the main program where the subroutine names are listed in parenthesis. All subroutines are adopted from SURF 3.2. In addition to these two types of files, another input file called input.par consists of beach orientation, peak wave period offshore, wind direction, wind speed and output interval. Similar to SURF 3.2, beach orientation angle is defined as the compass heading towards beach. The output interval represents the alongshore distance between adjacent output. It is noted that all input files are in MKS unit, i.e. in meter etc.

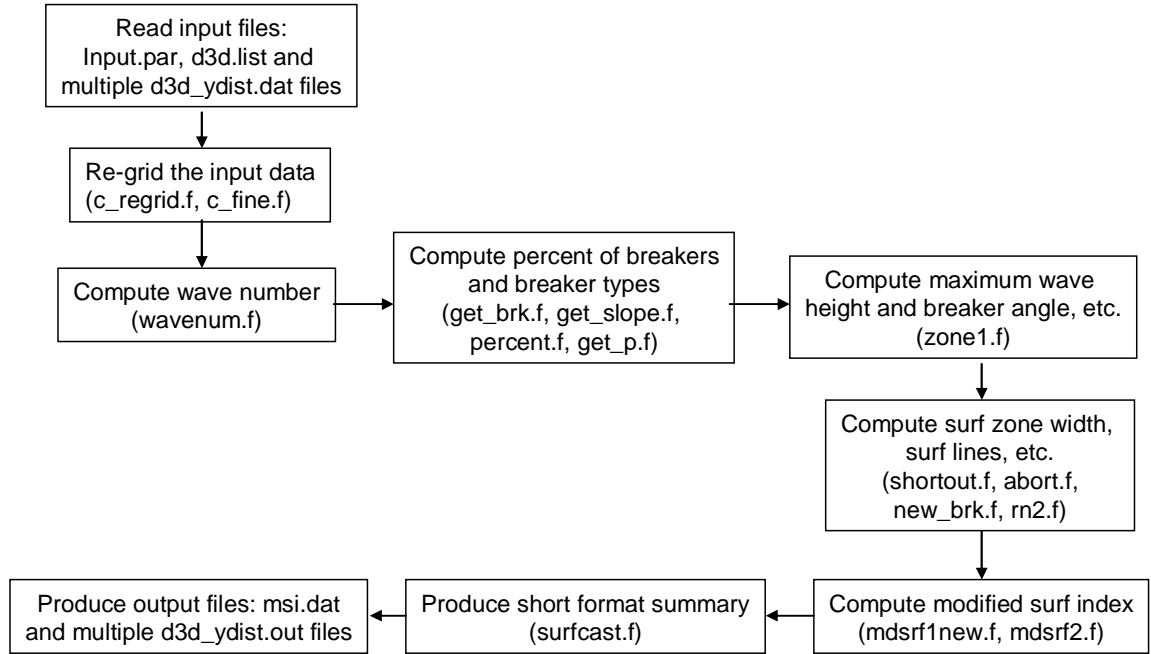


Fig. 2 – Flow chart for main Fortran program

Two types of output files are generated by this software. The first type of files is named with an “out” extension:

d3d_0.out
d3d_30.out
d3d_60.out
d3d_90.out
.....

They correspond to the surf summary for alongshore locations from those input files as listed in d3d.list, e.g. d3d_540.dat is for summary for input d3d_540.dat. Identical in content to the SURF 3.2 summary as listed in Appendix A, a sample “out” file from Delft3D results is listed here:

```

*****  *****  Coded Surf Forecast Follows  *****  *****

Significant Breaker Height          alfa =    4.9 ft
Maximum Breaker Height              bravo =    7.5 ft
Dominant Breaker Period              charlie =   8.0 sec
Dominant Breaker Type                delta = Spilling Surf
( 98% Spilling,   2% Plunging,   0% Surging)
Breaker Angle (toward left flank)    echo =   12.9 deg
Littoral Current (toward left flank)  foxtrot =    1.7 kts
Number of Surf Lines                golf1 =     4.2
Surf Zone Width                     golf2 =  538.0 ft
Average Wave Length                  =  126.7 ft
Wind Speed                          hotel1 =   10.0 kts
Wind Direction                      hotel2 =   46.0 deg

Modified Surf Index =                7.2

```

Units are in feet as in the SURF 3.2. The second type of output file is called msi.dat consisting of MSI, surf zone width and it’s alongshore distance. Many other surf parameters can easily be added on the output list.

4. SAMPLE RESULTS

A simulation of Delft3D was conducted for a Duck94 case at 1600 EST, 10 October 1994. The Delft3D output was processed by the new software. Fig. 3 shows the computed MSI (from the output file: msi.dat) as a function of longshore locations. The straight line at MSI = 8 represents operational limit of landing craft mechanized (LCM-6, LCM-8) and landing craft, air cushion (LCAC).

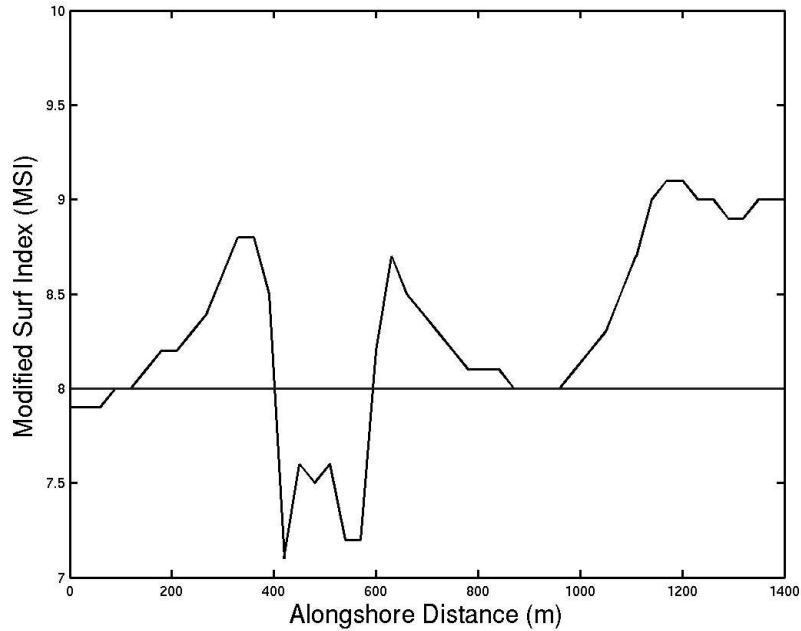


Fig. 3 – Sample plot of MSI for a Duck94 test case.

In Fig. 4, the edge of surf zone, derived from the computed surf zone width, is plotted with longshore current over depth. The white line corresponds to locations where 10 percent of waves are breaking. The yellow arrow-head represents 1 knot (0.51 m/s) of current.

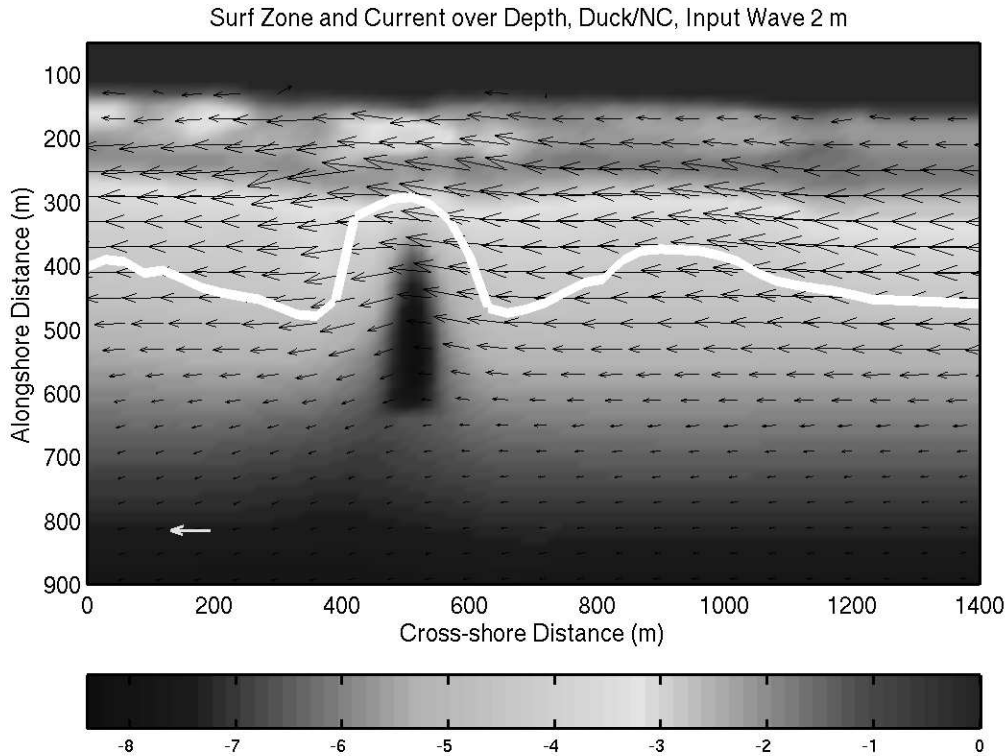


Fig. 4 – Sample plot showing the edge of surf zone with longshore currents over depth.

5. FUTURE PLANS

The present software applies to Cartesian grids where the coastline is almost straight, so that beach angle remains about the same for the entire modeling region. Future refinement of the software should extend its application to curvilinear grids. The present operational Perl script, for running Delft3D using DIOPS (Allard, et al., 2005) output, should also incorporate the present software, so that standard Navy surf output can be generated automatically.

ACKNOWLEDGEMENTS

This work was sponsored by SPAWAR PMW-150 under project: Nearshore wave and surf prediction. We thank Mr. Ted Mettlach, formerly at Neptune Sciences Inc., for evaluating the software.

REFERENCES

Allard, R., J. Dykes, J. Kaihatu, D. Wakeham, 2005: DIOPS: A PC-based wave, tide and surf prediction system, AMS Sixth Conference on Coastal Atmospheric and Oceanic Prediction and Processes. San Diego, CA.

- Delft Hydraulics, 2005: User manual Delft3D-FLOW, Delft Hydraulics, 614pp.
- Dykes, J.D., Y.L. Hsu, J.M. Kaihatu and R.A. Allard, 2005: Development of methodology and software for operational Delft3D applications, NRL Mem. Report (NRL/MR/7320-05-8832), 33pp.
- Elias, E.P.L., D.J.R. Walstra, J.A. Roelvink, M.J.F. Stive, M.D. Klein, 2000: Hydrodynamic validation of Delft3D with field measurements at Egmond, Proc. Coastal Eng., 2714-2727.
- Hsu, Y.L., T.R. Mettlach and M.D. Marshall, 2002: Validation test report for the Navy standard surf model, NRL formal report, NRL/FR/7322-02-10008, 28pp.
- Hsu, Y.L., J.M. Kaihatu, J.D. Dykes, R.A. Allard, 2006: Evaluation of Delft3D performance in nearshore flows, submitted as NRL Mem. Report.
- Mettlach, T.R., M.D. Marshall and Y.L. Hsu, 2002: Software design document for the Navy Standard Surf Model Version 3.2, NRL mem. report, NRL/MR/7320-02-8289, 184pp.
- Morris, B.J., 2001: Nearshore wave and current dynamics, Ph. D. Dissertation, Naval Postgraduate School.
- Roelvink, J. A., and G. K. F. M. van Banning , 1994: Design and development of DELFT3D and application to coastal morphodynamics, in Hydroinformatics, edited by A. Verwey et al., pp. 451– 456, A. A. Balkema, Brookfield, Vt.

APPENDICES

Appendix A – Sample SURF 3.2 output showing the standard Navy surf output

Test Case 1 Output File - case1.out

```
*****      *****  Surf Forecast  *****      *****

Navy Standard Surf Model      Version 3.2
Date and Time of Forecast:    01/01/2001    0100
Session Logged to file case1.out
Landing Zone Name             = case 7 beach
Sight Line                    =    0.0 deg
Equilibrium Beach Sediment    = medium sand
Wave Input Depth              =   25.0 ft
Sea  Height, Period, Direction =    0.0 ft,  0.0 sec,   0.0 deg
Swell Height, Period, Direction =    3.0 ft, 10.0 sec, 160.0 deg
Wind Speed                    =   10.0 kts
Wind Direction                =   240.0 deg
Tide Level                    =   -1.0 ft

Internally Generated Spectrum Used
Starting Depth                =   24.0 ft
Output Interval               =    5.0 ft
Computational grid spacing =   2.0 ft
Significant Wave Height Offshore =    3.0 ft
Peak Period                   =   10.0 sec
Average wave direction        =   -20.0 deg
Percent of Breaking Waves is less than 5.0 % at starting depth.

*****      *****  Coded Surf Forecast Follows  *****      *****
Significant Breaker Height          alfa =    3.6 ft
Maximum Breaker Height              bravo =    5.5 ft
Dominant Breaker Period              charlie =   10.0 sec
Dominant Breaker Type                delta = Spilling Surf
( 84% Spilling, 16% Plunging,  0% Surging)
Breaker Angle (toward left flank)    echo =   11.9 deg
Littoral Current (toward left flank)  foxtrot =    1.5 kts
Number of Surf Lines                golf1 =    2.0
Surf Zone Width                     golf2 =   246.0 ft
Average Wave Length                  =   120.9 ft
Wind Speed                          hotel1 =   10.0 kts
Wind Direction                      hotel2 =   240.0 deg

Modified Surf Index =              7.2
```

```
*****      *****  Detailed Surf Output Follows  *****      *****

Indx   Dist   Water   Sig Brkr   Max Brkr   Prcnt   Brkr   Littoral
      Offshore Depth   Height   Height   Brkng   Angle   Current
          (ft)   (ft)   (ft)   (ft)   waves   (deg)   (kts)

1     302.4     9.3   3.57   5.46    5.2   -12.7    0.12
2     297.4     9.1   3.57   5.47    5.5   -12.6    0.07
3     292.4     9.0   3.57   5.47    5.9   -12.5    0.01
.....
```

Appendix B – MATLAB delft3d2ascii.m to read Delft3D NEFIS format output files

```
%function [] = delft3d2ascii (runid,datetime,ctitle,roller)
% Read the Delft3D com and trim files to produce plots and ascii output
files
% Developed by E. Rogers, J. Dykes and L. Hsu, Code 7322, NRL
% The following parameters are placed here to be tailoured to your
% particular problem and preferences.
path(path, '/home/utah/hsu/delft3d/2.11.00/d3dmatlab')
runid='du8';
datetime='1994101016'
ctitle='Duck94';
roller =1; %when roller is used in *.mdf file, otherwise roller=0
% When roller is used, wave height should be derived from trim instead
com file

thin_s = 4; % plot thinning of arrows in current plot
thin_w = 4; % plot thinning in arrows in wave plot
quality = '75'; % jpeg quality level, affects figure size and quality
hscale = 75; %wave direction arrow scale
vscale = 125; %velocity arrow scale
vlegend = 0.5; %arrowhead legend velocity
range = [-18:3:0]; %depth contour interval and range
sc = 0.25; % contour smoothing, fine grid uses a small number, coarse
grid uses a large number
fz = 12; %font size

% For cropping
%xstart = 50; % staring index
%nxend = 171; % ending index

% Com file is opened and relevant info is extracted.
filename = sprintf ('%s%s', 'com-', runid, '.dat')
Nfs = vs_use (filename,'quiet');
filename2 = sprintf ('%s%s', 'trim-', runid, '.dat')
Nfs2 = vs_use (filename2,'quiet');

%vs_disp (Nfs); % show all stored parameters
%ntcurs = vs_get (Nfs,'KENMNT','NTCUR','quiet');
ref_date = vs_get (Nfs, 'PARAMS', 'IT01', 'quiet');
ref_time = vs_get (Nfs, 'PARAMS', 'IT02', 'quiet');
tscale = vs_get (Nfs, 'PARAMS', 'TSCALE', 'quiet');
timcur = vs_get (Nfs,'CURTIM','TIMCUR','quiet');
if iscell (timcur) == 0
    ntime = timcur;
else
    ntime = cell2mat (timcur);
end
tau = tscale * ntime(1)/3600;
cref_date = num2str (ref_date);
cref_time = num2str (ref_time);
yyyy = str2num (cref_date (1:4));
mm = str2num (cref_date (5:6));
dd = str2num (cref_date (7:8));
```

```

hh = str2num (cref_time);
ntime = datenum (yyyy, mm, dd, hh, 0, 0) + tau/24;
[vyyyy, vmm, vdd, vhh] = datevec (ntime);
valid_date = sprintf ('%d%0.2d%0.2d' , vyyyy, vmm, vdd);
%CLH valid_hour = sprintf ('%0.2d', vhh);
%somehow getting date and time from above is not always correct, so use
datetime (an input from script) instead
valid_date=datetime(1:8);
vhh =str2num(datetime(9:10));
yymmddhh=datetime(3:10);
valid_hour = sprintf ('%0.2d', vhh);

% Get currents
%data = vs_get (Nfs,'CURTIM','U1','quiet');
data = vs_get (Nfs2,'map-series','U1','quiet');
nt = size (data, 1); % nt is for getting the last data set
% if only one data set (one time step) used, the data array is no
longer a cell array.
if iscell (data) == 0
    USPD = data';
else
    USPD_CELL = data(nt);
    USPD = USPD_CELL{1,1}';
end
data = vs_get (Nfs2,'map-series','V1','quiet');
%data = vs_get (Nfs,'CURTIM','V1','quiet');
if iscell(data) == 0
    VSPD = data';
else
    VSPD_CELL = data(nt);
    VSPD = VSPD_CELL{1,1}';
end

% Get wave parameters
% when using roller, get wave energy, then convert to wave height

if roller == 1

data = vs_get (Nfs2, 'map-rol-series', 'EWAVE1', 'quiet');
if iscell (data) == 0
    weng = data';
else
    weng_cell = data(nt);
    weng = weng_cell{1,1}';
end
% Change engery to wave height
whrms=sqrt(8*weng/9.81/1025);

else

whrms = vs_get (Nfs, 'WAVTIM', 'HRMS', 'quiet')';

end ; % end of roller logic

wdir = vs_get (Nfs, 'WAVTIM', 'DIR', 'quiet')';
DP0 = vs_get (Nfs, 'INITBOT', 'DP0', 'quiet')';
X2D = vs_get (Nfs, 'GRID', 'XCOR', 'quiet')';

```

```

Y2D = vs_get (Nfs, 'GRID', 'YCOR', 'quiet');

% Get rid of extra row and column put in by Delft3D
nx = size (X2D, 1) - 1;
ny = size (X2D, 2) - 1;
DP0 = -DP0(1:nx, 1:ny); % set depth as negative
X2D = X2D(1:nx, 1:ny);
Y2D = Y2D(1:nx, 1:ny);
x=X2D(:,1);
y=Y2D(1,:);
USPD=USPD(1:nx,1:ny);
VSPD=VSPD(1:nx,1:ny);
whsig=sqrt(2)*whrms(1:nx,1:ny);
wdir=wdir(1:nx,1:ny);

% Crop boundaries from m = nxstart to nxend
%CLH no cropping nx = nxend - nxstart + 1;

%Y2D = Y2D(nxstart:nxend, :);
%X2D = X2D(nxstart:nxend, :);
%USPD = USPD(nxstart:nxend, :);
%VSPD = VSPD(nxstart:nxend, :);
%DP0 = DP0(nxstart:nxend, :);
%whrms = whrms(nxstart:nxend, :);
%wdir = wdir(nxstart:nxend, :);

% thinning
it = 0;
%skip near the boundary
jskip = 0;
for i = 1:thin_s:nx
    it = it + 1;
    jt = 0;
    for j = (jskip+1):thin_s:(ny - jskip)
        jt = jt + 1;
        Y2D_s(it, jt) = Y2D(i, j);
        X2D_s(it, jt) = X2D(i, j);
        USPD_s(it, jt) = USPD(i, j);
        VSPD_s(it, jt) = VSPD(i, j);
    end
end
it = 0;
for i = 1:thin_w:nx
    it = it + 1;
    jt = 0;
    % for j = (jskip+1):2*thin_w:(ny - jskip)
    for j = (jskip+1):thin_w:(ny - jskip)
        jt = jt + 1;
        Y2D_w(it, jt) = Y2D(i, j);
        X2D_w(it, jt) = X2D(i, j);
        whx(it, jt) = whrms(i, j) * cos (wdir(i, j) * pi/180);
        why(it, jt) = whrms(i, j) * sin (wdir(i, j) * pi/180);
    end
end

figure (1), clf, hold off
axis equal

```

```

h = pcolor (x, y, DP0');
axis equal
set (h,'linestyle','none')
shading interp; % smooth looking
%axis ([min(x) max(x) min(y) max(y)])
newmap=jet;
%lightjet=newmap(12:58,:,:)
colormap(jet);
caxis ([min(min(DP0)) 0])
colorbar
xlabel ('x (m)','fontsize',fz)
ylabel ('y (m)','fontsize',fz)
title1 = sprintf ('%s%s%s%s%s%s', 'Current over Depth, ', ctitle, '
', valid_date, ' ', valid_hour, 'EST');
title ([title1] , 'fontsize',10)
set (gca,'fontsize',fz)
hold on
quiver (X2D_s, Y2D_s, USPD_s*vscale, VSPD_s*vscale, 0, 'k-');
set (findobj('type', 'line'), 'linewidth', 0.03)
hold on
xd = (max(x)-min(x))/10+min(x);
yd = (max(y)-min(y))/16+min(y);
xd2 = (max(x)-min(x))/11+min(x);
yd2 = (max(y)-min(y))/11+min(y);
% current legend
quiver(xd,yd,vscale*vlegend,0.,0,'m-')
text(xd2,yd2,'0.5 m/s','fontsize',11,'color','m')

figure (2)
axis equal
h = pcolor (x, y, whsig');
axis equal
set (h,'linestyle','none')
shading interp; % for smooth looking
colormap(jet)
colorbar
caxis([0 max(max(whsig))])
xlabel ('x (m)', 'fontsize', fz)
ylabel ('y (m)', 'fontsize', fz)
title2 = sprintf ('%s%s%s%s%s%s', 'Sig Wav Hgt and Depth Cont, ',
ctitle, ' ', valid_date, ' ', valid_hour, 'Z');
title ([title2] , 'fontsize',10)
set (gca, 'fontsize', fz)
hold on
quiver (X2D_w,Y2D_w,whx*hscale,why*hscale,0,'k-'); % wave vector
set (findobj('type','line'),'linewidth',0.03)
hold on;

xf = linspace (min(x), max(x), nx*sc);
yf = linspace (min(y), max(y), ny*sc);
[X,Y] = meshgrid (x,y);
[XI,YI] = meshgrid (xf,yf);
ZI = interp2 (X, Y, DP0', XI, YI, 'cubic');
[C,h] = contour (xf, yf, ZI, range, 'm--');
set (h, 'linewidth', 0.03)
hh=clabel(C,h,'LabelSpacing',216);
set (hh, 'color', 'm','fontsize', 11)

```

```

str = ['print -f1 -dpng ' 'vduc' yymmddhh];
disp (str)
eval (str)
%str = ['print -f2 -dpng ' 'w' datetime ];
str = ['print -f2 -dpng ' 'wduc' yymmddhh ];
disp (str);

% If the coordinates were in UTM, this tabular output would be suitable
for ingest into ArcView.
%fileout = sprintf ('%s%s%s', ctitle, 'table', '.dat');
%fout = fopen (fileout, 'w');
%fprintf (fout, '%9s %9s %9s %9s %9s %9s %9s\n', 'X', 'Y', 'Depth(m)',
'Hsig(m)', 'Dir(Cart)', 'U', 'V');
%in = 0;
%for iy = 1:ny
%   for ix = 1:nx
%       if (DP0(ix,iy) < 0)
%           fprintf (fout, '%9.1f %9.1f %9.1f %9.1f %9.1f %9.4f %9.4f\n',
X2D(ix,iy), Y2D(ix,iy), DP0(ix,iy), whsig(ix,iy), wdir(ix,iy),
USPD(ix,iy), VSPD(ix,iy));
%       else
%           fprintf (fout, '%9.1f %9.1f %9.1f %9.1f %9.1f %9.4f %9.4f\n',
X2D(ix,iy), Y2D(ix,iy), DP0(ix,iy), NaN, NaN, NaN, NaN);
%       end
%   end
%end
%fclose (fout);

fid1=fopen('d3d.list','w');

for jj = 1 : 2: ny
ydist=15*(jj-1);

% change cartesian to surf wave angle convention
    tmpwdir(:,jj)=-(wdir(:,jj)-180);
    for ii=1:nx ;
% fix for wave angle beyond computation, i.e. on land
        if (tmpwdir(ii,jj) == 180);
            tmpwdir(ii,jj)=0.0;
        end
    end

%only save the data when depth is larger than 0, i.e. wet
ztmp=-DP0(:,jj);
L=find(ztmp > 0);

output= [(flipud(X2D(L,jj)))'; (flipud(-
DP0(L,jj)))';(flipud(whsig(L,jj)))';
(flipud(VSPD(L,jj)))';(flipud(tmpwdir(L,jj)))'];
profileout=sprintf ('%s', 'd3d_',num2str(ydist), '.dat');
fid=fopen(profileout,'w');
fprintf(fid,'%8.2f %8.2f %8.2f %8.2f %8.1f\n ',output);
fclose(fid);
fprintf(fid1,'%s\n ',profileout);
end
fclose(fid1);

```


Appendix C - Main Fortran Program surfdel3d.f

(All subroutines are adapted from SURF 3.2 and are not listed here)

```
C This program is developed to produce standard Navy surf parameters
C from Delft3D
C All subroutines are adapted from SURF 3.2.
C Programmed by Larry Hsu and validated by Ted Mettlach
C For questions, contact Larry Hsu, code 7322, NRL,
C hsu@nrlssc.navy.mil

C Input files are produced from delft2ascii.m (MATLAB): d3d.list and
C d3d_ydist.dat files. Additional input file is input.par which
C provides
C beach orientation, peak wave period offshore, wind direction, wind
C speed
C and output interval.

C Output files are msi.dat and d3d_ydist.out files which give surf
C summary.

c local variables from SURF 3.2:
c alfa real      significant breaker height
c bravo real     maximum breaker height
c chrlie real    dominant breaker period
c dangle real    angle between directional bins
c depname char*40 depth profile file name
c dsea real      input direction for sea contribution
c dstart real    input starting depth
c dswell real     input swell direction for internally
c                generated spectrum
c dxy1(points) real corresponding depths with no tide
c echo real      breaker angle
c ehsig real     significant wave height from directional
c                spectrum
c esowm (dirnum,freqnum) real directional wave spectrum
c file_dat char*40 output file name *.dat
c file_in char*40 input filename
c file_out char*40 output file name *.out
c file_tmp char*40 temporary file
c file_spc char*40 output file name for refracted wave spectrum
c foxtrt real    longshore current speed and direction
c fracname char*40 wave refraction file name
c freq (freqnum) real input wave spectrum center frequencies
c freq_rs(freqnum) real frequencies assoc'd w/ input refracttion &
c                shoal matrices
c freq1 (freqnum) real beginning frequency bin values
c freq2 (freqnum) real ending frequency bin values
c gamma2 real    beach orientation, compass leading directly toward beach
c golf1 real     number of surf lines
c golf2 real     surf zone width
c hsea real      input significant wave height for sea
c                contribution to pierson moskowi
c hswell real     input significant wave height for
c                internally generated spectrum
c iday integer   input day
```

```

c idirec integer number of direction bins in the input spectrum
c idirec_rs integer number of directions assoc'd w/ input
c refraction & shoal matrices
c ifreq integer number of frequency bands in the input spectrum
c ifreq_rs integer no. of freqs assoc'd w/ input refraction
c & shoal matrices
c igamma integer beach orientation rotated 90° from
c original heading toward beach
c ihour integer input hour
c ihtl1 real wind speed coded surf forecast value
c ihtl2 real wind direction
c imin integer input minute
c imonth integer input month
c iyear integer input year
c jgamma integer temporary value set to beach orientation
c line char*80 temporary variable used to read lines
c lin_stress logical longshore current solution (true or false)
c lndname char*40 input landing zone name
c nnn integer number of points in the input depth array
c pct(4) real percent of different breaker types
c pct (1) = spilling
c pct (2) = plunging
c pct (3) = surging
c pct (4) = total
c period(freqnum) real period array (1/frequency)
c psea real input wave period for sea contribution to
c pierson spectrum
c pswell real input swell period for internally generated
c spectrum
c roller logical roller usage (true or false)
c self_st char*1 self start flag (yes or no)
c slope real bottom slope
c spectra logical does input spectra exist? (true or false)
c spefile char*40 selected wave spectrum file name
c spedepth real depth at input wave spectrum
c surfy logical significant wave heights greater than
c 0.5 ft? (true or false)
c tide real input tide level
c wdir real input wind direction compass heading
c from which wind comes
c wspd real input wind speed
c xcoeff(dirnum, freqnum) real shoaling coefficient matrix adjusted to
c xfrom and freq
c xcoeff_rs(dirnum, freqnum) real shoaling coefficient matrix input from
c fracname
c xdelt real surf zone output interval
c xdelt_gr real self-adjusting cross-shore grid step
c xfrom(dirnum) real direction array, direction from which wave energy
c comes
c xfrom_rs (dirnum) real direction array assoc'd w/ input
c refraction & shoal matrices
c xtheta(dirnum, freqnum) real refraction angle matrix adjusted to xfrom
c and freq
c xtheta_rs(dirnum, freqnum) real refraction angle matrix input from
c fracname
c xx1(points) real adjusted cross-shore distances from depth profile
c ydepth char*1 input depth profile used? (yes or no)

```

```

c ydetail char*1    detailed output? (yes or no)
c yrefrac char*1    is refraction considered in analysis? (yes or no)
c ystr  char*1      is strait coast refraction used? (yes or no)
c
    program surfdelft3d
c
    implicit none
    include 'common.inc'
c
    integer ifreq, idirec, nnn, igamma
    integer ifreq_rs, idirec_rs
    integer iyear, imonth, iday, ihour, imin, jgamma
    integer spe_type
c
    real xx1(points), dxy1(points), xfrom(dirnum), pct(4)
    real xfrom_rs(dirnum)
    real freq(freqnum), period(freqnum),
freq1(freqnum),freq2(freqnum)
    real freq_rs(freqnum)
    real esowm(dirnum, freqnum), xcoeff(dirnum, freqnum)
    real xcoeff_rs(dirnum, freqnum)
    real xtheta(dirnum, freqnum)
    real xtheta_rs(dirnum, freqnum)
    real alfa, bravo, chrlie, echo, foxtrt, golf1, golf2
    real ehsig, dangle, slope, gamma2, dstart, xdelt, xdelt_gr
    real hsea, psea, dsea, hswell, pswell, dswell
    real wspd, wdir, tide, ihtl1, ihtl2, spedepth
c
    character*1 ydepth, ydetail, self_st, yrefrac, ystr
    character*40 file_in, file_out, fracname, depname, file_dat
    character*40 file_tmp, spefile, file_spc
    character*40 lndname
c
    logical roller, lin_stress, spectra, surfy
    real tmp(points), dum1, dum2
    real distmax
    real fqd, per, dp
    real dxy(points)
    real xtemp(points), xktemp(points), htemp(points), ptemp(points)
    real thetatemp(points), thetamin, thetamax
    real ebtemp(points), along(points), blong(points), v(points)
    real clong(points), rk(points, 4), bl(points)
    real sum1, width, hlmax, h2max, vmin, vmax
    real df, ftsq2msq, temp
    integer iimax, wid_ii, nnn1, nnn2, nnn3, nnn4
    real hrms, xk
    integer j_ii, j, k , i , j_ii2
    real xshift
    character*40 file_in2
    real tmp_h(points), tmp_theta(points), tmp_v(points)
    real xxin(points), zzin(points)
    real xkd, l0, xoff,p(4),dum
    logical surf, brk10
    integer ii, ndepth, fend, irun, iline, iydist
    character*20 tmpfile(80), cydist, cdum
    real tmpmsi(80), tmpzone(80), srfmod, tmp_pct(80,4), tmpydist(80)
    real xmin, tmpxmin(80)

```

```

c setting model options
    data roller, lin_stress /.true., .false./
c
*****
c setup forecast, open input and output files
c
    if (debug ) write(*,*) 'SURF'

    ydetail  = 'y'
    surfy    = .true.
    surf     = .true.
    brk10    = .false.
    j_ii     = 1
    ihtl1    = wspd
    ihtl2    = wdir
    xshift   = 0.0
    xdelt_gr = 2.0*dcal

cccccccccccccccc
    open(41, file = 'd3d.list', status = 'old')
    do irun = 1,80

        read(41,'(a)', end=42) file_in
        write(*,*)file_in
        if (irun .gt. 0.0) then
cccccccccccccccc

c        write(*,*) 'Enter Input File Name: '
c        read(*,'(a)') file_in
c        file_in = 'd3d_99.dat'

c
    i=0
    j=0
    do while (i .eq. 0)
        j=j+1
        if(file_in(j:j) .eq. '.') then
            fend = j
            i=1
        endif
    enddo

c
    file_out = file_in(1:fend-1)//'.out'
c    file_dat = file_in(1:fend-1)//'dat.dat'
c    file_in2 = file_in(1:fend-1)//'par.dat'
CLH
    cydist=file_in(1:fend-1)

    read(cydist, 'a4,i4')cdum, iydist
    write(*,*) 'ydist=', iydist

c    ydist=iachar(file_in(5:fend-1))

C    file_in2 = 'D101016.dat'
    file_in2 = 'input.par'

c    write(*,*)'file_in  = ',file_in

```

```

c      write(*,*)'file_out = ',file_out
c      write(*,*)'file_dat = ',file_dat
c      write(*,*)'file_in2 = ',file_in2

c      iunit=20 in common.inc

      open(iunit, file=file_out, status='unknown')
c      write(*,*) file_out,' OPENED'

c      open(30, file=file_dat, status='unknown')
c      write(*,*) file_dat,' OPENED'

      open(11, file=file_in2, status='old')
c      write(*,*) file_in2,' OPENED'

      read(11,*) gamma2, per, wspd, wdir, xdelt
      close (11)
c      write(*,*) file_in2,' CLOSED'

      fqd=1/per
      call c_gamma(gamma2, igamma)
      jgamma = igamma

      close (10)
      open(10, file = file_in, status='old')
c      write(*,*) file_in,' OPENED'

      j_ii = -1

      do j=1,1000
        read (10,*,end=111) xxin(j),zzin(j),tmp(j),
tmpv(j),tmptheta(j)
        tmptheta(j) = tmptheta(j) * pi / 180.0
c        write(*,*)xxin(j),zzin(j),tmp(j), tmpv(j),tmptheta(j)
        ndepth=j
      enddo
111  continue
      close (10)
CLH
      xmin=xxin(ndepth)
c      write(*,*) ndepth

C re-grid to even spaced
      call c_regrid(ndepth,xxin,zzin,      xdelt_gr,nnn1,xx1,      dxy)
      call c_regrid(ndepth,xxin,tmp,      xdelt_gr,nnn2,xx1,      tmph)
      call c_regrid(ndepth,xxin,tmptheta,xdelt_gr,nnn3,xx1,thetatemp)
      call c_regrid(ndepth,xxin,tmpv,      xdelt_gr,nnn4,xx1,      v)

c      write(*,*)'nnn = [' ,nnn1,', ' ,nnn2,', ' ,nnn3,', ' ,nnn4,']'
      nnn = nnn1

      do ii=1,nnn
        dp = dxy (ii)
c        get wave number
        xkd = 0.01
        call wavenum(fqd,dp,xkd)
        xk = xkd

```

```

        l0 = 2.0*pi / xk
        hrms = tmpb(ii)/1.42
c      write(*,*) dxy(ii), tmpb(ii), hrms, thetatem(ii), v(ii)

        xoff = xx1(ii)-xshift
c
c      write(*,*) 'xoff = ',xoff
c      write(*,*) 'xshift = ',xshift

        call get_brk(ii, nnn, xx1, dxy, xdelt_gr, hrms, l0,
+          per, xoff, rk, bl, brkl0, distmax, p)

        xtemp(ii) = xoff
        xktemp(ii) = xk
        htemp(ii) = hrms
        ptemp(ii) = p(4)
        iimax = ii

c      write(*,*)'xoff =', xoff
c      write(*,*)'xk =', xk
c      write(*,*)'hrms =', hrms
c      write(*,*)'p(4) =', p(4)
c      surf zone width index
c      if (j_ii .lt. 0 .and. p(4) .ge. 10)j_ii = ii

        enddo

c      write(*,*)'j_ii = ',j_ii,'iimax = ',iimax
c calculate surf zone parameters. print to output file.
c
        call zonel(j_ii, iimax, dxy, xtemp, htemp, ptemp, thetatem,
+ xktemp, v, distmax, vmax, vmin, thetamin, thetamax,
+ sum1, width, j, k, hlmax, h2max, wid_ii)

c      write(*,*)'width = ',widthc
c      write(*,*)'hlmax = ',hlmax
c      write(*,*)'h2max = ',h2max

c      |      |      |
c      15      16      18
c      write(*,*)'k = ',k, 'width = ',width,'sum1 = ', sum1
c
c calculate short output values
c
        call shortout(wdir, wspd, j, iimax, dxy, xtemp, sum1, k,
+hlmax, h2max, per, pct, thetamin, thetamax, vmax, vmin,
+width, igamma, bl, rk, htemp, wid_ii,
+jgamma, alfa, bravo, chrlie, echo, foxtrt, golf1, golf2,
+ihtl1, ihtl2)

c      write(*,*) 'echo = ', echo

c      write short output

        call surfcast(pct, depname, lndname, slope, ydepth, alfa,
+bravo, chrlie, echo, foxtrt,golf1, golf2,
+ihtl1, ihtl2)

```

```

c      write(*,*) 'echo   = ', echo

      write(*,*) 'alfa    = ', alfa
      write(*,*) 'chrлие = ', chrлие
      write(*,*) 'pct     = ', pct
      write(*,*) 'echo    = ', echo
      write(*,*) 'foxtprt = ', foxtprt
      write(*,*) 'jgamma  = ', jgamma
      write(*,*) 'ihtl1   = ', ihtl1
      write(*,*) 'ihtl2   = ', ihtl2
c      write modified surf index
CLH add MSI
      call mdsrflnew (alfa, chrлие, pct, echo, foxtprt, jgamma,
+ihtl1, ihtl2, file_out,srfmod)
      tmpzone(irun)=golf2
      tmpxmin(irun)=xmin
      tmpydist(irun)=iydist
      tmpmsi(irun)=srfmod
      do i=1,4
      tmpcpt(irun,i)=pct(i)
      enddo
      iline=irun
      tmpfile(irun)=file_in
      write(*,*) 'MSI   = ', srfmod
      close(30)
CLH      if (ydetail .eq. 'y') call prt_out3(file_dat)

      close(iunit)
c
c end program surf.for
endif
42      continue
      enddo
      close(41)
      open(25, file='msi.dat')
      do ii=1, iline
C      write(25, "(a,i3,2f7.1)") tmpfile(ii), ii, tmpmsi(ii),
tmpzone(ii)
      write(25, "(7f7.1)" ) tmpydist(ii),tmpmsi(ii),tmpxmin(ii),
+ tmpzone(ii),tmpcpt(ii,1), tmpcpt(ii,2), tmpcpt(ii,3)
      end do
      close (25)
      stop
      end

```